XCELL

THE NEWSLETTER FOR XILINX PROGRAMMABLE GATE ARRAY USERS

Issue 3 Second/Third Quarter 1989

This is a combined issue for the second and third quarter of 1989. The editing of the revised '89 Data Book interfered with this newsletter, but the 4Q issue will again be on time in November.

As you may have seen in our address, Xilinx has moved to new quarters, about four miles south of the previous location. It is a substantially larger building with enough space to accomodate the expected rapid growth of the coming years.

This issue of XCELL describes two important software updates:

ADI 2.2 is shipping right now, and the new Design Manager, XDM, will ship in October. Both programs are sent, free of charge, to all our customers with an active update subscription.

This issue also gives you a glimpse of our next generation devices, the XC4000 family, to become available in the second quarter of next year.

Peter Alfke, Editor

Table of Contents

XC3020s Break World Speed Record

Many designers use our LCAs as glue-logic, MSI, or PAL replacement, others use them to build customized peripherals or control subsystems. Some designers also take advantage of the reprogrammability of our devices, but nobody seems to have gone as far as Jean Vuillemin, Patrice Bertin, and Didier Roncin at the Paris Research Laboratory of Digital Equipment Corporation.

They liked the LCA architecture so much that they conceived and built a VME-bus based universal accelerator board with a 5 x 5 array of XC3020s plus 64K x 64 of static RAM. This board is non-specialized; its function is determined solely by the LCA configuration.

One interesting application is in data encryption. The well-known RSA (Rivest-Shamir-Adleman) public key encryption algorithm is easy to use and virtually unbreakable. The security of this code is based on the fact that it is easy for the originator to generate the product of two very large prime numbers, but impossible for the unauthorized interceptor to derive these factors, when only their product is known.

This method of encryption requires a simple mathematical operation: a exp b modulo c, made difficult only by the fact that a, b, and c are 512-bit long binary numbers. The accelerator board configures the LCAs into a 512-bit long serial multiplier, running at a 25 MHz clock rate. This design achieves a 32 kbps throughput rate, while the fastest software-based designs are limited to less than 2 kbps, and the two fastest publicly known specialized ASIC designs (one of them classified) run at less than 13 kbps.

This general purpose LCA-based design outperforms specialized hardware by a factor of 2.5, with-out using any illegal drugs!

And, according to Jean Vuillemin, it only takes a man-week to go from algorithm to working hardware, which means that you can try out a variety of different approaches and enhancements, without exceeding your budget in money or time.

We are convinced that more and more users will explore these hidden potentials of our re-configurable logic with similar dramatic results.

PA

The limitation is not in the silicon, it's in the user's imagination!

Component Availability (September 1989)

		48 PIN		68 PIN		84 PIN		100 PIN		132 PIN		164 PIN	175 PIN	
		PLASTIC DIP	CERAMIC DIP	PLASTIC PLCC	CERAMIC PGA	PLASTIC PLCC	CERAMIC PGA	PLASTIC PQFP	CERAMIC CQFP	PLASTIC PGA	CERAMIC PGA	CERAMIC CQFP	PLASTIC PGA	CERAMIC PGA
		-PD48	-CD48	-PC68	-PG68	-PC84	-PG84	-PQ100	-CQ100	-PP132	-PG132	-CQ164	-PP175	-PG175
XC2064	-33	C	I M	C I	C I M									
	-50	C	I	C I	C I M									
	-70			C I	C I									
	-100			C *	C *									
XC2018	-33			C I		C I	C I M B							
	-50			C I		C I	C I M B							
	-70			C I		C I	C I							
	-100			C *		C *	C *							
XC3020	-50			C I		C I	C I M B	C I *	C I M B					
	-70			C I		C I	C I	C I *	C I					
	-100			C *		C *	C *	C **	C **					
XC3030	-50			C I		C I	C I M	C I *	C I M **					
	-70			C I		C I	C I	C I *	C I **					
	-100			C *		C *	C *	C **	C **					
XC3042	-50					C I	C I M	C I *	C I M *	C I **	C I M B			
	-70					C I	C I	C I *	C I *	C I **	C I			
	-100					C *	C *	C **	C *	C **	C *			
XC3064	-50									C I **	C I M			
	-70									C I **	C I			
	-100									C **	C **			
XC3090	-50											C I M	C I	C I M B
	-70											C I	C I	C I
	-100											C **	C **	C **

* Sample—Now Production—4Q89

** Sample—4Q89 Production—1Q90

Current Software List

The following is a list of the current software revision levels for Xilinx's development system products. This is a listing of the diskettes currently being shipped with each of the development system products, as of September 1, 1989.

DS21 XACT ver. 2.20
XACT ver. 2.12
DOS 16/M Loader ver. 2.49
XC3030/XC3042 die files
XC3020-PC84 package file
XC3042-PG132 package file
Speeds file update: 5/19/89
CQ164, CQ/PQ100 package files

DS22 P-SILOS (16K) ver. 2.20
P-Silos ver, 3C. 8–16K
xnf2silo ver. 2.12

DS221 P-SILOS (5K) ver. 2.20 (fomerly DS122)
P-Silos ver. 3C.8–5K
xnf2silo ver. 2.12

DS23 ADI ver. 2.21

ADI ver. 2.21
Speeds file update: 5/19/89

DS23-AP1 ver. 2.21
DS23-AP1 ver. 2.21

DS23-SN1 ADI ver. 2.21
DS23-SN1 ver. 2.21

DS28 XACTOR ver. 2.10
XACTOR 2.10

DS31 FUTURENET DASH INTERFACE ver. 2.20
FutureNet interface and library,
PIN2XNF ver. 2.21

DS311 FUTURENET TTL LIBRARY (formally DS 40)
FutureNet TTL Library ver. 1.0

DS32 SCHEMA II+ INTERFACE ver. 2.20 REV2

DS33 DAISY INTERFACE (DNIX) ver. 1.04

DS34 MENTOR INTERFACE ver. 2.10
Mentor IDEA interface&library ver. 2.10

DS35 OrCAD/SDT INTERFACE ver. 1.0
OrCAD/SDT interface and library ver. 1.0

DS52 SCHEMA II+ AND ADI ver. 2.21
Schema II+ ver. 2.21
Xilinx/Schema interface ver. 2.21
DS23 ADI ver. 2.21

DS 51 SCHEMA II+, ADI, AND XACT ver. 2.21
DS52 ver. 2.21
DS21 XACT ver. 2.20

DS54 DASH-LCA and ADI ver. 2.21
DASH-LCA ver. 4.10d
PIN2XNF ver. 2.21
DS23 ADI ver. 2.21
DASHEVAL

DS53 DASH-LCA, ADI, +XACT ver. 2.21
DS54 ver. 2.21
DS21 XACT ver. 2.20

DS81 SERIAL CONFIGURATION PROM PROGRAMMER
SCP programmer ver. 2.00

COMING SOON:
Versions of the MAKEBITS/MAKEPROM program for the Apollo and SUN3 workstations will be released soon. These programs will allow a user to generate a bitstream from an LCA file on the workstations. (MAKEBITS/MAKEPROM is part of the XACT package on the PC.)

DS501-API XACT Dev. System ver. 2.21 on Apollo

DS501-SNI XACT Dev. System ver. 2.21 on SUN3

DS112 ENHANCED SERIAL CONFIG WAFER PROM PROGRAMMER
This programmer has adjustable and programmable voltage to support future Xilinx serial PROMs.

Xilinx Introduces Design Manager

The Xilinx LCA development process, from design entry to design verification, involves several different programs which must be invoked in the proper sequence with the correct options. The new Xilinx Design Manager simplifies this development process by automating the design-to-LCA translation, and providing on-line help and well-organized menus to guide the user through the entire LCA design flow.

The Xilinx Design Manager (XDM) is a menu-based interface that "looks and feels" much like the XACT design editor. All programs and options appear as menu items which can be selected with the mouse or entered on the command line. The menus are grouped according to development stages: design entry, design translation, placement and routing, and design verification.

The on-line HELP system included with the XDM will be appreciated by even the most experienced LCA designer. A helpful description of each program and program option can be displayed at any time.

In addition to on-line help and a convenient menu display, the Xilinx Design Manager offers automatic design translation. A single program called XMAKE automatically invokes all the programs needed to convert your schematics and Boolean equations into an LCA file. You no longer need to remember which programs to invoke and when to invoke them. Instead, let XMAKE do all the work.

The XMAKE program generates and executes a MAK file similar to the UNIX "makefile". The MAK file lists all of the programs and options used to translate your design into an LCA file. Since this file can be edited and re-executed, the sophisticated user has complete control over the design translation process. There is no need to execute each translation program individually, although that capability still exists.

The XMAKE program has several options which make it easy to use the new logic partitioning features of ADI 2.21 (the Automatic Design Implementation software). In a hierarchical design, there are three logic partitioning schemes:

1. Combine all levels of hierarchy and then partition the "flattened" logic into CLBs and IOBs. The resulting design will be dense, but may be difficult to route.
2. Partition the logic in hierarchical blocks which contain the "FILE=" parameter, and then merge this partitioned logic with the rest of the design. The resulting design will be less dense, but should be easier to route.
3. Partition the logic in ALL hierarchical blocks and then merge the partitioned logic with the rest of the design. The resulting design will be even less dense but should be much easier to route.

Each of these partitioning schemes can be implemented with a single XMAKE program option. In a very short time, and without any changes to schematics or PDS files, you can translate your design into an LCA file three different ways and then choose the best solution. Without XMAKE, this same procedure would take considerably more time and effort, and would require changes to your schematics and PDS files.

The Xilinx Design Manager is both a user-friendly and powerful tool. It makes the LCA development process simpler to understand and easier to complete and will be welcomed by all LCA designers. Currently registered customers will receive their Design Manager updates in October free of charge.

CAL

DS23 ADI Version 2.21

The version 2.21 upgrade of the DS23 Automated Design Implementation (ADI) package has been sent to all currently registered DS23 owners. This revision of ADI has improved many existing features, while adding powerful new capabilities. The result is a better partitioned design and improved placement and routing algorithms. Some highlights of the new ADI software are given below:

1. Logic partitioning can be directed from the schematic editor, if so desired. The CLBMAP, a new mapping symbol, can be included on a schematic to control the mapping of the design into the configurable logic blocks (CLBs) of the LCA architecture. Of course, the XNF2LCA program still provides automated mapping of the design.
2. Further support is provided for hierarchically-structured design entry. If desired, portions of a design can be independently mapped into the logic and I/O blocks of an LCA prior to merging them into the top-level design file. This often eases the placement and routing of large, complex designs. The format of the XNF file has been updated to support hierarchical design. The result is greater flexibility for the designer and more efficiently-partitioned designs. The changes to the XNF specification also will aid in the development of interfaces to timing simulators.
3. The Automatic Placement and Routing (APR) program has been improved, particulary for large and/or sparse designs. Additional support for "incremental" design changes is provided (that is, easy methods for making small logic changes to designs that have already been placed and routed).

BKF

Xilinx Unveils the XC4000 Family

For many months, our customers have expressed a growing interest in the next generation of Programmable Gate Arrays. Just as microprocessor users have come to expect advance information about the next generation of CPUs, the users of Programmable Gate Arrays are eager to hear about future plans for the next generation of Logic Cell Arrays. Until now such revelations would have been premature. Details were still being finalized, and product availability was too far in the future. Now we are ready to indicate what will become available in mid-1990 as the XC4000 family. We are still necessarily vague about specific details, as we are still working on some of the patents covering key features of the new architecture, but we think that future users should hear in general terms what kind of building blocks will be available from Xilinx. Holding back all information until silicon is available would be a disservice to the community of users who need time to plan the right strategy for their next designs. Early information about the silicon and software applications is also needed to provide third party software support for design entry and design verification.

Technology is evolutionary

The XC4000 will again be based on CMOS SRAM technology. Configuration data will be shifted into the device and will be stored in CMOS latches, similar to the XC2000 and XC3000 parts. But the manufacturing process will use more aggressive sub-micron design rules. This means smaller chips with higher speed, shorter delays, and ultimately lower cost.

Logic density is increased

XC4000 will be a family of devices with logic densities from 2000 to 20,000 gates, more than double the present XC3090's. This complexity range covers 90% of all gate array designs forecast for the mid 1990s.

Speed is enhanced

Improved architectural features will result in a 50% performance improvement over the XC3000 family. Semiconductor processing advances should provide another 30–40% increase in speed. The product of these factors indicates that the XC4000 family will be useful at system clock rates up to twice those possible with the recently introduced 100 MHz version of the XC3000 family.

Versatility is increased

The structure of the XC4000 CLB is similar to the XC3000 CLB. There are two flip-flops, each with a 4-input function generator. But the inputs to each of the function generators in the XC4000 CLB are completely independent. Not only eight, but even nine variables can drive one CLB and can be combined to generate an output or drive a flip-flop. One CLB can thus implement any two independent functions of four variables, and many functions of eight and nine variables are possible, like parity generators and two-bit adders or accumulators. Both flip-flop outputs and combinatorial function outputs are available simultaneously to the general interconnect network.

RAM is available

The XC4000 devices have on-chip high-speed static RAM, up to 2,500 bytes for the largest member of the family, less for the smaller family members. This RAM can be used to store variables, to act as registers, shift registers, FIFO or LIFO buffers, multiple accumulators, look-up tables, etc.

Wide decoding is fast

Decoding of 10- to 32-bit wide address fields was cumbersome and slow in the XC2000 family, but improved with the long lines in the XC3000 family. The XC4000 family goes one step further and provides wide address decoding of up to 50 inputs or internal signals with a speed equivalent to 15 ns PALs.

Counters are compact and fast

The versatility of the CLBs makes it possible to design fast and compact counters. The XC3000 family achieves 25 MHz speed and one bit per CLB for an 8-bit synchronous presettable counter. The XC4000 family beats this with 50 MHz and 2 bits per CLB for the same design. Presettable as well as non-presettable, and up or down as well as up/down counters, 4- to 16-bits long, will be available as part of an extensive macro library.

Arithmetic is simpler and faster

Adders and subtractors using either ripple-carry, carry generate/carry propagate, or conditional sum algorithms are faster and smaller than in the XC3000 family. A 16-bit adder can run at 50 MHz.

Pipelining speeds up the system

The abundance of free flip-flops invites pipelined design techniques for highest throughput speed. Dividing a complex function into several parts and executing them in sequence and in parallel can multiply system performance.

Abundant routing resources

Compared to the XC2000 and XC3000 families, routing resources have been increased dramatically. There are more than twice as many vertical and horizontal long lines, and more local interconnects. Not only are there more routing resources, they are also more accessible to the CLBs, and the access is more regular, less specialized. This makes it easier for the software to complete the routing of complex interconnect patterns. Our software designers worked intimately with our chip architects to specify a structure that is not only powerful and flexible, but also easy for the software to utilize. The goal is to provide automated push-button software that routes almost all designs, even densely populated ones, automatically. But we still give the designer the option to get involved in the partitioning,

placement and even routing if that is meaningful in order to optimize performance.

New and improved development systems

The software for the XC4000 family uses new data structures optimized for larger devices, and new algorithms for better utilization and automatic design implementation. Xilinx now has more R&D effort in software design than in chip design.

Improved algorithms for logic partitioning, placement, and routing will result in fully automatic implementation for most designs, and will also reduce the time required for design completion. Improved data structures will improve both performance and the efficiency of memory usage.

The development system will include a large macro library, including hard macros with optimized layouts for high performance.

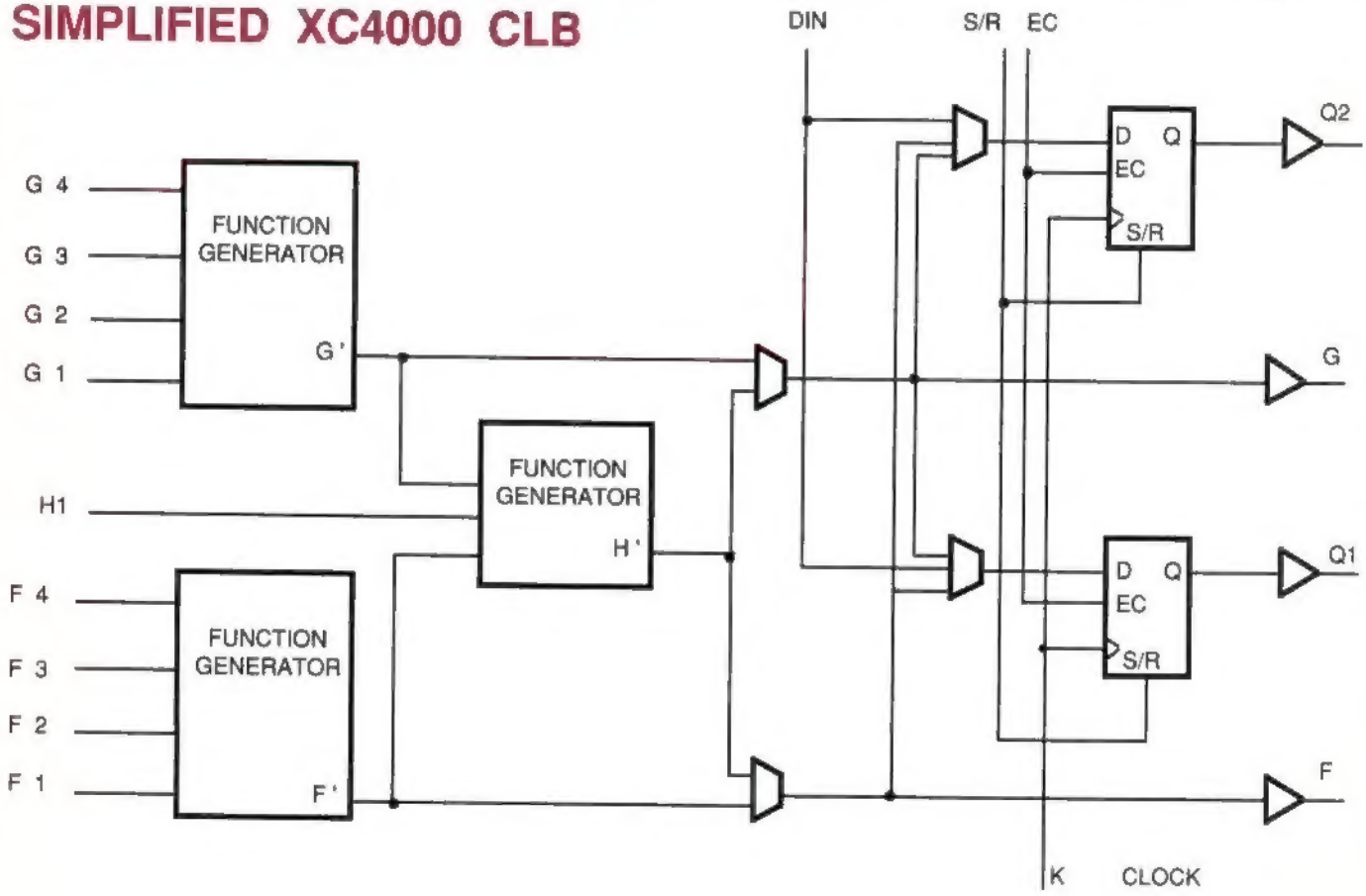
PC support will be for the 80386 and newer processors that avoid the segmented-memory limitations of the older Intel processors. We will port the development software onto different CAE workstations, Sun and Apollo will be among the first workstations supported.

Conclusion

The XC4000 family will extend programmable gate arrays to the 20,000 gate level. Gate utilization, speed and versatility will be enhanced, and the automated placement and routing (APR) software will provide push-button solutions.

PA

SIMPLIFIED XC4000 CLB



Performance Comparison	3000 Family	4000 Family
- 8/9 Bit Parity Generate/Check	14 ns/2 CLBs	5 ns/1 CLBs
- 24 Bit Input Decode	20 ns/5 CLBs	10 ns/0 CLB
- 32:32 Bit Identity Comparator	16 ns/16 CLBs	12 ns/8 CLBs
- 16:1 Multiplexer	28 ns/10 CLBs or 21 ns/7CLBs	10 ns/5 CLBs
- 16 Bit Synchronous Counter	35 ns/14 CLBs	20 ns/8 CLBs
- 16 Bit Loadable Up/Down Counter	60 ns/18 CLBs	20 ns/8 CLBs

Automatic Placement and Simulated Annealing

The Automatic Placement and Routing (APR) program is used during design implementation to perform the placement and routing of an LCA design. The APR program reads an LCA file, generates a new block placement, routes the nets of the design, and writes the result to another LCA file. This article describes the basics of the placement algorithm used by APR, including a description of the options available to the user for controlling the placement process.

Determining the optimal placement of the CLBs and IOBs of an LCA design is a very demanding problem. The number of possible block placements is astronomical, even for the smallest LCA devices, and the number of routing alternatives for a given placement is equally astronomical. Thus, a brute force exhaustive search of all possible placements is impractical.

Within the APR program, a "simulated annealing" algorithm is used to determine the optimal block placement. "Annealing" refers to the crystallization process in which a metal is melted and then slowly cooled back to freezing in order to form highly-ordered crystals with few defects. "Simulated annealing" is an algorithm for finding good solutions to complicated optimization problems (such as the automatic placement of gate arrays) in a manner analogous to the physical crystallization of a metal.

During each iteration of the simulated annealing placement algorithm, two or more blocks are exchanged at random, and the "routability" of the resulting placement is calculated. The routability is expressed in terms of a "routing score", calculated using a formula that includes the length of the routes, the net weighting, and the number of available routing channels as factors. LCA designs are not actually routed during the placement process. The new placement can then be accepted or rejected before going on to the next iteration; if accepted, the new placement becomes the starting point for the next iteration. Accepting only those placements that result in improved routability might cause the design to be trapped in a local optimum, which might be significantly worse than the real optimum.

Therefore, a number of factors determine if a new placement is accepted or rejected. The simulated annealing placement algorithm has two phases: the annealing phase and the quenching phase. During annealing, a new placement that results in better routability is always accepted. However, if the new placement is worse, there is still a probability that it will be accepted, depending on how much worse it is and the current "temperature". The worse the new placement, the less likely it will be accepted; the higher the temperature, the more likely it will be accepted. A random number seed, taken from the system's time-of-day clock by default, also affects the calculation of this probability. As the algorithm proceeds, the design is slowly "cooled" by lowering the temperature. At high temperatures, blocks move freely; as the temperature is lowered, blocks tend to settle into good locations. Once the design is sufficiently cooled, the quenching phase is entered, wherein only better placements are accepted.

By default, APR automatically calculates and assigns a starting temperature sufficient to "melt" the design, that is, a temperature high enough to allow total scrambling of the initial placement. The algorithm also automatically determines the rate at which the temperature is lowered and when the design has cooled sufficiently to allow quenching.

Information describing the progress of the placement algorithm is displayed on the screen while APR is executing, including the temperature, cooling rate, and routing score for the current iteration. The first several iterations typically show a drastically decreasing temperature and a somewhat stable average placement score. As the design cools, the temperature will decrease more slowly and the average and best scores will start to fall. The standard deviation of the scores is the main determinant of how fast the temperature will decrease. As the design nears final placement (temperatures < 100), the temperature will begin to fall more rapidly again.

Command line options specified during the invocation of the APR program can be used to control the placement process, if so desired, as described below:

Option: `-B temp`

Meaning: Set Beginning Temperature

The default is calculated by the APR program, and will be high enough to destroy the placement of the input LCA file. Setting it higher than this makes no sense. But if the initial placement is fairly good, a low starting temperature will prevent the final placement from being radically different from the initial placement. Hence, the temperature can be set to a low number (~100) when the designer

wants to limit movement during annealing in order to preserve the structure of the original placement.

Option: `-K rate`

Meaning: Set Cooling Rate (betwen 5 and 50)

The cooling rate is the percent of temperature decrease from one iteration of the placement algorithm to the next. By default, APR dynamically computes the cooling rate for each iteration. A high cooling rate (~40) can be used for a "quick-and-dirty" placement, especially for sparse designs.

Option: `-Q`

Meaning: Quench Only

This option skips the annealing phase, thereby only allowing placements that are better than the input design. This option is useful for designs whose initial placement is very good, and will result in a significant reduction in APR execution time.

Option: `-E temp`

Meaning: Set Ending Temperature

The default is 1. Lower ending temperatures should not be used. Raising the ending temperature will cause quenching to occur sooner, decreasing execution time but degrading the final placement results. Raising the ending temperature is another means of obtaining a "quick-and-dirty" placement for a sparse design.

Option: `-R seed`

Meaning: Set the Random Number Seed (1 to 32767)

By default, the seed is derived from the system's time-of-day clock. Successive runs of APR without the -R option will produce different results. Users are encouraged to execute APR several times (using the APRLOOP program), and then choose the best design from the resulting placements.

BKF

The Tilde De-Mystified

Timing values given by XACT or APR are sometimes preceded by the symbol ~, called Tilde by its Spanish name, but really meaning "approximately". How should the user interpret this symbol?

All non-tilde timing values given by XACT or APR are carefully simulated, modeled, and measured worst-case values, guaranteed over the range of processing tolerances and temperature and supply voltage variations. The user can have confidence that no device will ever exceed these values.

The tilde is a disclaimer. It means that the delay is generated by so many concatenated resistor (or pass-transistor) -capacitor elements, that our design and test engineers have less confidence in the accuracy of the model and the repeatability of the timing value. Xilinx cannot guarantee it as an absolute worst-case value.

The number following the tilde is still a conservative specification; most likely the parameter in question is better than this value. But there is not the same guarantee as there is with non-tilde values.

What is the user to do?

If a "tilde-value" is critical to your design, you have two choices:

1. Change the lay-out or routing such that the long uncertain delay is broken up into two "non-tilde" values, either by passing the net through a BIDI or through an unused CLB, or by dividing the net into two branches.
2. Add 25% to the value and ignore the tilde, making the reasonable assumption that this factor 1.25 compensates for the modeling uncertainty.

XC2064 and XC2018 ACLK delay values, though below 10 ns, are sometimes preceded by a tilde. You can safely ignore the tilde in these cases.

There has been a misleading explanation that the tilde indicates propagation delay differences between the rising and the falling edge of a signal. This is not true. Different from original 2.0µ technology XC2000 parts, all new 1.2µ technology devices, and especially the XC3000 family parts, have their delays finely balanced. Our designers have painstakingly adjusted n-channel and p-channel geometries to achieve driving impedances and threshold voltages that guarantee virtually identical propagation delays for rising and falling transitions.

Maybe we have been overly pessimistic and caused unjustified concern with the tilde. But we prefer to be cautious and make a distinction between worst-case guaranteed values and intelligent, albeit conservative, estimates.

PA

The Secrets of "Tie"

After creating a completely placed and routed LCA design, you must convert the design into a bitstream in order to download it to the LCA. This conversion is done using the **Makebits** utility in XACT. When you select the Makebits command inside the Makebits menu, you will see three choices: **Tie, Norestore** and **Verbose**. For prototyping, you don't need any of these options; selecting **Done** will generate a bitstream. The Verbose option is only needed when you want the program to send longer messages to the screen.

Always run the **DRC** program before running Makebits. DRC checks for electrical errors in the design. Fix all fatal errors and verify all warnings. Some warnings are acceptable. For instance, if a CLB is configured to output a ground signal, it has an output but no inputs. In this case, DRC will issue a warning which can be ignored.

When you are ready to generate a bitstream for production, you should select the **Tie** and **Norestore** options. **Tie** connects all unused inputs and interconnects in the device to legitimate logic levels in order to reduce power consumption and on-chip noise. Tie begins by configuring unused CLBs to create a ground net and then tie these nets to as many places as possible. Any input or interconnect that cannot be tied to these ground nets will be tied to any other existing net unless the net is flagged critical. The **Norestore** option temporarily saves the timing data for the tied design. *Tying a design is not necessary for prototyping but is necessary for production.*

Tying a design may increase the delay of some nets, but usually only by a nanosecond or two. After tying a design, choose the **Norestore** option to save the timing information and then examine the timing of critical nets using the **QueryNet** command under the **Misc** menu to see if the tie operation affected them adversely. If it has, go back to the XACT **Editlca** program and flag the particular net critical with the FLAGNET command under the **Net** menu. This prevents the tie operation from tying to this net and increasing its delay.

The **QueryNet** command has an option called **Tiechange** which compares net timing before and after tie. Unfortunately, this option is not working in XACT version 2.12. It will be fixed in the next release.

After executing a **Makebits Tie Norestore** command, the timing data must be restored to its non-tied state before one can exit the MakeBits program. The tied design is better electrically, but the many added, logically redundant tie connections make the design very difficult to analyze and virtually impossible to edit. It is, therefore, advisable to store the untied LCA file for future reference. Use the **Restore** command under the **Misc** menu to return the LCA file information to its original state.

You can also save the tied file by using the **Save** command in the XACT Executive. You will be warned that the timing data has not been restored when you exit Makebits and warned that the interconnect is tied down when you try to save the file.

Makebits Tie Problems

Tie Failure

If the Makebits Tie program cannot completely tie down a design, it will abort and display messages about what it could not tie. To get a hard copy of these error messages, use your computer's print screen or output file redirect command. Tie often fails because nets are unnecessarily flagged critical. Never flag the global or alternate clock nets critical since these nets are never affected by tie. Go back into XACT Editlca and remove all critical net flags from the design by flagging the nets uncritical and then run Makebits Tie Norestore again. Check to make sure that Tie did not affect the timing adversely. If it did, go back to XACT Editlca and flag critical only those nets whose timing was affected, and then run Makebits Tie again.

Sometimes flagging nets uncritical is not enough. In very rare cases, Tie cannot find a way to connect to some interconnect segments. This some-times affects interconnects near IOB .Q pins in the 3000 family LCAs because they cannot be used to source tie nets. If a certain interconnect (usually a Programmable Interconnect Point or PIP) cannot be tied, Makebits will list the PIPs's location in a format such as:

```
col.A.local.13:row.I.local.10
```

This location can be found in the XACT Editlca editor by typing:

```
find col.A.local.13:row.I.local.10
```

Once the location is found, use the Editnet command to route any nearby net to the PIP that could not be tied, save the design and run Makebits Tie again.

If after removing critical flags and manually editing untied PIPs, the design still does not tie completely, call technical support for assistance.

Stackstop Errors

XACT version 2.12 cannot tie XC3090 designs that are almost empty. If you try to tie such a design, you will probably get a stackstop error from Makebits because Makebits Tie caused a tie net to be routed to so many places that it overflows the 64K stack on the PC. This error will not occur on the Sun or Apollo and will be fixed in the next release of the PC version (late '89).

If you get this error, remember that Tie is only required for production designs. If you are still prototyping, we ask that you be patient and wait for the new release. If you want to go to production with a design that gives you this error and you cannot wait for the new release, call your local sales representative and arrange to have the design sent in for Xilinx to tie at the factory using Sun Workstations. You may be asked the question why you want to go to production with a virtually empty XC3090, but Xilinx will gladly tie the design if that is your wish.

TCW

Recover from System Crash or Accidental Erasure

One of the most horrifying moments in your life as an LCA designer comes when a power glitch interrupts XACT after you completed hours of work which you had not saved. Even more agonizing is the accidental erasure of an LCA design. These accidents need not ruin your day since the designers of the XACT software have provided ways for you to recover your unsaved work.

If your PC crashes during an XACT editing session, don't panic. Reboot your system and return to the directory you were working in. In the directory you will find the LCA file you were working on. It will be complete up to the point of your last save. You will also find a file with the same name as your LCA file but with a .DBK extension. This is the "data back-up" file and it contains all of the commands you executed in XACT between the last save and the crash of the program. Use the Execute command in XACT and this file to restore your work. First edit the DBK file and remove all commands that caused errors and all exits to DOS. Then call up XACT and load your design into the editor. Once it's loaded, click on the Misc menu and then select the Execute command. You will be asked for the name of the file to execute. Type the name of the data back up file and the .DBK extension and then press ENTER. XACT will then execute all of the commands in the file. When the file is finished executing, the LCA design file will be back to the state it was just before the crash. Save the design immediately!

If you accidentally erase an LCA design file, you can recover your work in a way similar to what is described above. When you save an LCA design file, XACT also saves the previous version of the design in a file of the same name but with a .ODF (old design file) extension and it creates a log file containing all the commands executed between the previous save and this save. The log file has the same name as the LCA file but with a .LOG extension. Use this file to restore your work. First edit the file and remove all commands that caused errors and all exits to DOS. Then call up XACT and load the LCA file into the editor. Once it's loaded, click on the Misc menu and select the Execute command. You will be asked for the name of the file to execute, type the name of the log file with the .LOG extension and press ENTER. XACT will then execute all of the commands in the file. When the file is finished executing, the LCA file will be back to the state of the file that was erased. Save the design immediately and often thereafter!

It is prudent to save LCA design files often and to back them up on floppy disks, but should the unthinkable happen, XACT provides a good way to get you back to work quickly.

TCW

50 Macros Added to the DS311
(Formerly DS40)

Name	Description	# of CLBs	Speed in MHz -70	-100
X74160U	Loadable BCD Up-Counter	6	31	38
X74160D	Loadable BCD Down-Counter	6	31	35
X74161U	Loadable Hex Up-Counter	5	33	41
X74161D	Loadable Hex Down-Counter	5	33	41
X74165S	Synch. Loadable 8-Bit Shift Register	4	55	71
X74165A	Asynch. Loadable 8-Bit Shift Register	8	55	71
C16UPLD	Loadable 16-Bit Up-Counter	19	17	20
C16DNLD	Loadable 16-Bit Down-Counter	19	17	20
C8UDLD	Loadable 8-Bit Up/Down Counter	9	18	22
C16UDLD	Loadable 16-Bit Up/Down Counter	18	11	15
SAR	Successive Approximation Register	*1	25	38
BRM	Binary Rate Multiplier	* 1.5	34	50
X7474	Flip-Flop with Asynch. Preset & Clear	2	55	71
C3SQUARE	Divide-By-Three with 50% Duty Cycle	2	27	33
C5SQUARE	Divide-By-Five with 50% Duty Cycle	2	27	33
M4-1C	Four-to-One Multiplexer in One CLB	1	**8ns	**7ns
BRLSHFT4	Four-Bit Barrel Shifter	4	**17ns	**13ns
CBINRIP	Binary Ripple Counter	2	50	62
CDECRIP	BCD Ripple Counter	2	50	62
C5BIT32	5-Bit Divide By 32 Shift Register Counter (Also Divide by 31, 30,...9)	3	38	55
C3BIT8	3-Bit Divide By 8 Shift Register Counter (Also Divide by 7, 6, 5, and 4)	2	50	55
C3BIT8O7	3-Bit Divide By 8 or 7 Shift Register Counter	2	50	55
PHFRCOMP	Phase/Frequency Comparator	2	*** ≥ 10	

* Per Bit (Plus 1 CLB for the Controller for SAR)
** Propagation Delay
*** Based on actual breadboard results. This macro requires 3 IOBs.

DASH Users: Don't Use DCM to Merge Drawings

Complex LCA designs should be entered in a hierarchical manner, where "modules" on high-level drawings are fully described in lower-level drawings. When entering designs with the DASH schematic editor, the DCM program should not be used to merge the hierarchical drawings.

The translation of a DASH schematic drawing into its corresponding Xilinx netlist file involves three steps. First, the DCM program creates a "drawing connectivity model" from the drawing file. Next, the PINC utility translates the .DCM file into a pinlist file. Lastly, PIN2XNF is used to translate the FutureNet pinlist into the XNF (Xilinx Netlist File) format. (Note: The DCM and PINC utilities are supplied by Data I/O Inc. as part of the FutureNet DASH schematic editor, and are used to generate a FutureNet-compatible netlist from the drawing files. The PIN2XNF program is supplied by Xilinx to translate FutureNet netlists to the XNF format.)

The DCM utility can be invoked in two different manners.

1. DCM can be run on a single-page drawing file by entering

   `DCM filename`

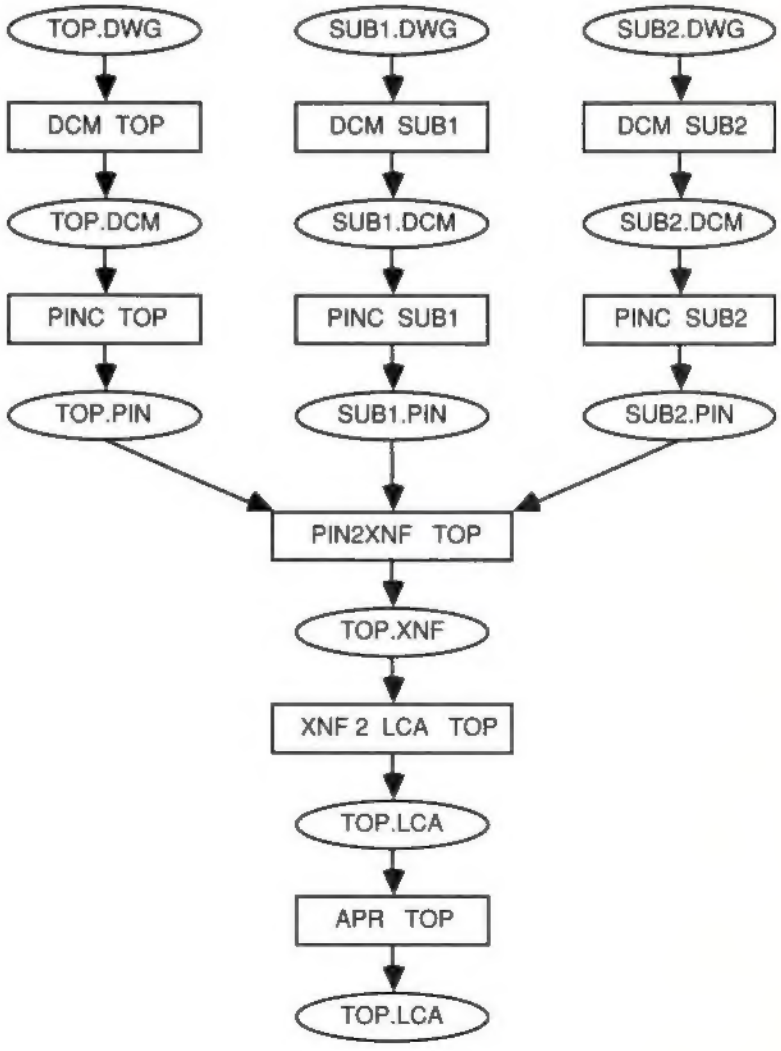
   from the DOS prompt. The .DCM file for that single drawing is then created.

2. Alternatively, if DCM is invoked without a filename, it will prompt the user for the name of the top-level "root" files and the names of all the lower-level drawing files. Using this method, multiple drawings of a hierarchically-structured design can be processed into a single .DCM file by the DCM utility. However, due to incompatibilities in the file structures, using DCM to resolve hierarchical references in this manner may result in unpredictable and undocumented errors from the PIN2XNF program, such as "Undefined pin type in file filename.PIN", and the creation of incorrect XNF files. Xilinx has no control over this, since DCM is a FutureNet program and not a Xilinx program. *This method is not recommended.*

However, this second method of invoking DCM can be used to process a multi-page DASH drawing. For example, if the top-level drawing spans several pages, the filename for each page is entered in response to the prompt for "root" files. However, don't cross hierarchical boundaries using DCM.

The correct methodology for resolving hierarchy for a design entered using DASH is to run DCM and PINC separately for each drawing file, and then let PIN2XNF resolve the hierarchical references. For example, the figure below illustrates the design flow for a design consisting of a top-level drawing named TOP.DWG that references two lower-level drawings, SUB1.DWG and SUB2.DWG.

BKF



No Can Do

Xilinx LCAs offer a wide range of design options and many system-oriented features. There are, however, some restrictions.

Here are the things you should not even try to do in the XC3000 family:

The on-chip input pull-up resistor cannot be used if the pin is configured as I/O, i.e., if the configuration allows the output to be activated. The resistor cannot be used to pull up a 3-stated output,use an external resistor instead.

Bidirectional buses are limited to the length of one Horizontal Long Line. There is no way to interconnect bidirectional buses. There is no pass-transistor between the buses, and two back-to-back amplifiers would latch up.

IOB flip-flops and latches can be reset only by the global RESET package pin that resets every flip-flop and latch on the chip.

Clock polarity is determined at the sources of the IOB's clock line, not at each individual IOB.

IOB latches driven from the same clock line as a flip-flop have a surprising latch enable polarity: Active Low latch enable if the flip-flop clocks on the rising edge, active High latch enable if the flip-flop clocks on the falling edge. This enable polarity must be specified explicitly to avoid a "fatal DRC error".

The two flip-flops in a CLB cannot have separate clocks, clock enable or asynchronous reset inputs.

The global clock distribution network cannot be used for anything else but driving CLB and IOB clock inputs. The alternate clock network, however, has limited access to the general purpose interconnects.

PA

Universal Prototyping Board for LCAs

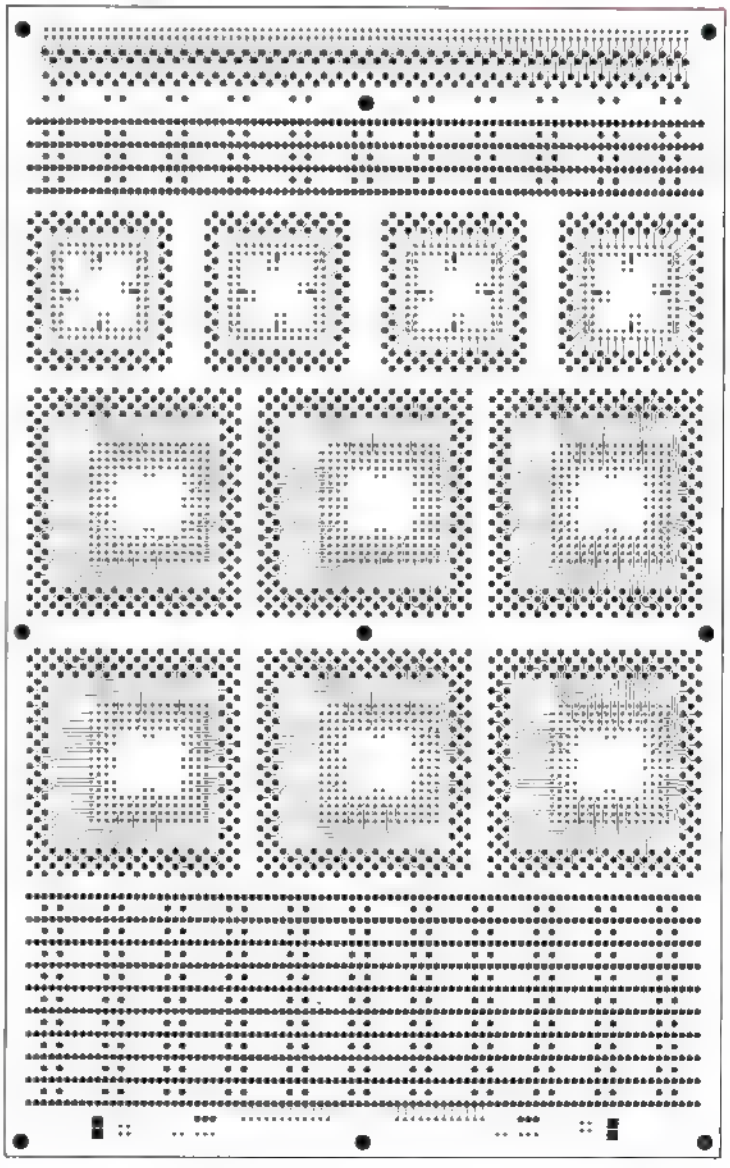
The UNILINX™ prototyping board from Kyros Corp. allows you to interconnect up to six XC3090 devices and up to four XC2064, 2018, 3020, 3030 or 3042 devices plus a large number of standard DIP components, all on one 9" by 15" four layer PC board with buried power and ground planes.

Designed by Kyros for its own prototyping work in the tele-communications field, the board uses an insulation displacement wiring system from BICC-VERO, called Speedwire. The UNILINX board comes fully loaded with speedwire pins. You use a special insertion tool to press 30 gauge wirewrap wire into the bottom end of any pin. On the top of the board the other side of the pin forms a socket ready to receive your component.

The UNILINX board costs $1800, fully pinned and socketed. For more information call (201) 838-8700, or write to Kyros Corp., 22 Park Place, Butler, NJ 07405.

Xilinx has not (yet) seen or used this board; we can, therefore, not endorse it, but we like to pass this information on to our readers, since several of you have asked for it.

Keith Mayers, Bill Heath PA



Mouse Problem with DASH-LCA?

Some users cannot get the Xilinx-supplied FutureNet Dash-LCA to work properly with their mouse, even though they have specified the mouse properly in their AUTOEXEC.BAT file.

The problem can be as simple as an extra "space" character in your AUTOEXEC.BAT file!

If you edit this file with a word processor, make sure that you do not enter extra spaces in the SET DMOUSE command, not even at the end of the line.

For example, if you are using a Mouse System mouse on the COM1: serial port, your AUTOEXEC.BAT file must include the line:

```
SET DMOUSE=S1
```

There must be only one space on the line, between SET and DMOUSE.

We recommend that you let the INSTALL program modify your AUTOEXEC.BAT file for you, when you first install the DASH-LCA software.

TCW

Need PLCC Sockets?

The Feb.16, 1989 edition of EDN (pages 63 to 72) has an exhaustive article on this subject. See also the list of socket manufacturers on page 2-115 in the 1989 Xilinx Data Book.

PA

Controlling the XC1736 Serial PROM

Most connections between the LCA and its Serial PROM are simple and self-explanatory:

- The DATA output of the 1736 (or 1736s) drives DIN of the LCA.
- The master LCA's CCLK output drives the CLK input of the 1736(s)
- The $\overline{CEO}$ output of any 1736 can be used to drive the $\overline{CE}$ input of the next 1736 in a cascaded chain of PROMs.
- VPP must be connected to VCC. Leaving VPP open can lead to unreliable, temperature-dependent operation.

There are, however, two different ways to use the inputs CE and OE.

1. D/$\overline{P}$ of the LCA drives both $\overline{CE}$ and $\overline{OE}$ in parallel. This is the simplest connection, but it fails when a user applies RESET during the LCA configuration process The LCA will abort the configuration and then restart a new configuration, as intended, but the 1736 does not reset its address counter, since it never saw a High level on its $\overline{OE}$ input. The new configuration will, therefore, read whatever data is stored at the higher address locations in the 1736. Unless the 1736 was programmed with this in mind, the re-configuration will fail.
2. The LDC output from the LCA drives the $\overline{CE}$ input of the 1736, while its $\overline{OE}$ input is driven by the inversion of the LCA's RESET input. This connection works under all normal circumstances, even when the user aborts a configuration before D/$\overline{P}$ has gone High. The High level on the $\overline{OE}$ input during RESET clears the 1736-internal address pointer, so that the reconfiguration starts at the beginning. Most designs have a "spare" inverter or inverting gate that can be used for this purpose.

PA

Need Help with Fine Pitch SMT?

Fine Pitch Technology
2216 Lundy Avenue
San Jose, CA 95131
tel. (408) 433-9550

They can help you with PC board design engineering, prototyping, and low volume production.

PA

FutureNet Lowers Price on DASH-LCA to DASH4 Upgrade

The upgrade is now priced at $695 (formerly $1995). Upgrading to DASH4 provides the capability to create schematics for a wide range of parts, in addition to LCA design. HP LaserJet and plotter support is included in the DASH4 upgrade.

Contact your local FutureNet sales office, or call FutureNet at (206) 881-6444.

PW

1989 Data Book is Available

We revised, enhanced and expanded the Data Book and gave it a 1989 identification on the cover and the spine. Everybody actively designing with Xilinx LCAs should get hold of this new version and take advantage of the additional information. Donate your old copy to a friend, for it is still full of relevant information.

Here is a summary of the most important changes:

Chapter 1

A few trivial errors were fixed.

Chapter 2

Page 2-10, Fig. 12 now properly connects the y outputs to the IOBs.

Page 2-12, Fig. 15 now uses the right caption under each part of the drawing.

Page 2-29 shows a new selection table.

Pages 2-33 through 2-37 now include all new package options.

Pages 2-40 to 43 not only include the -100 option, but also offer a completely revised description of the timing parameters. The new structure might answer many of your questions.

Pages 2-44 through 47 add -100 timing.

Pages 2-48 through 55 give all package dimensions.

The XC2000 data sheets are hardly changed, except for adding the -100 option.

The military data sheet is essentially new. Note that the XC2064 had been eliminated, but the XC3020B and XC3090B have been added.

The 1736 description on page 2-139 through 149 is essentially unchanged, but 2-158 describes the upcoming larger EPROM (1764).

Page 2-159 lists a large number of socket manufacturers.

Chapter 3

New updated reliability data on page 3-5 and radiation hardness data on page 3-10.

Chapter 4

Description of the new Training Course on page 4-7.

Chapter 5

Completely re-done, it offers more tutorial and a concise overview of the Xilinx development process and its many options. It introduces the new XDM design manager and various workstation options. Note that many development system options have new part numbers.

Chapter 6

The applications section was significantly expanded.

There are two pages of general design considerations for the XC2000 and 3000 families.

The IOB input set-up time is described better and the problem of minimum delay specification is explained on page 6-18.

Synchronized Start-up is described on pages 6-19.

Page 6-21 explains how metastability was measured, and page 6-22 gives practical circuits for battery back-up.

The next 40 pages are Application Briefs, some from the '88 Data Book, some from the previous editions of XCELL, plus some new ones:

- A 4-input multiplexer in one CLB.
- A 16 bit adder/accumulator running at 30MHz (!), twice as fast as the conventional circuit (6-30).
- A 30 MHz counter with synchronous reset (6-36).

Examples of unconventional LCA applications including a gigahertz presettable counter (using an inexpensive ECL prescaler) and a frequency/phase comparator.

Page 6-44 shows a complete 100 MHz frequency counter in one XC3020.

Two new state machine designs address high speed simple application, and slower, very complex state machines with up to 240 states and 128-way (!) branch capability.

Chapter 7

Adds an article reprint explaining the use of reconfigurability in several actual designs (7-24).

Chapter 8

Contains an updated index and an up-to-date sales office listing.

PA

Training Course Schedule

The Xilinx Programmable Gate Array Training Course is a comprehensive class covering the Logic Cell Array component architecture and Xilinx development systems, with emphasis on the XC3000 family devices. All courses are held at Xilinx headquarters in San Jose, CA. The tuition fee is $850 per student. To enroll, call the Training Administrator at Xilinx head-quarters (408-559-7778). Class size is limited, so early enrollment is recommended. The course schedule for the remainder of 1989 follows:

Sept. 11–14	**Nov. 6–9**
Oct. 16–19	**Dec. 4–7**

State Machines

Application Brief BY PETER ALFKE

State machine design is a methodology that defines the contents of all flip-flops for any possible state of the design, then defines all possible paths that can cause the design to go from one state to another. In its simplest form this is just a rigorous way of designing synchronous logic, like 4-bit counters. For complex designs, the state machine approach gives the designer a tool to investigate all possible operating conditions and avoid overlooked hang-up states or undesired transitions. Xilinx LCAs with their abundance of flip-flops lend themselves well to state machine designs.

SIMPLE, FAST STATE MACHINES

Using the 5-input function generator of the XC3000-70 family devices as a 32 bit ROM, a state machine with up to 32 states without any conditional jumps uses only 5 CLBs and operates at up to 50 MHz.

The 5 registered CLB outputs drive the 5 function generator inputs of the 5 CLBs in parallel. This implements a fully programmable sequencer similar to the synchronous counter shown in the left column of page 6-23.

For a smaller number of states, some inputs can be used as conditional jump inputs. Encoding these condition codes may require an additional level of logic which reduces the maximum clock rate to 30 MHz.

SIMPLE STATE MACHINE RUNS AT 30 MHz

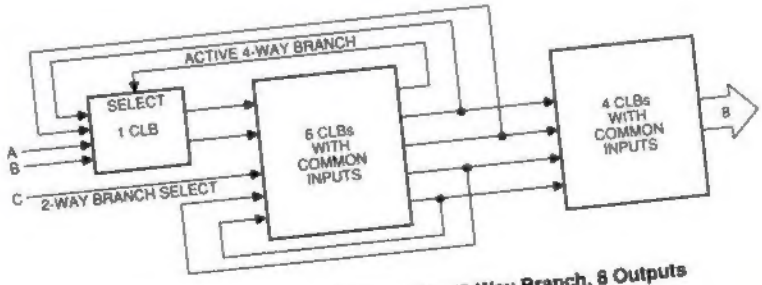
This simple state machine uses only eleven CLBs. It has up to 16 states, and eight outputs, each decoding/encoding any combination of states. It performs a 2-way branch from any state to any one of two freely assigned states, (possibly including the present state) determined by control input C. (Avoid the branch by making both destination states equal).

This design can also perform an 8-way branch from any state so programmed to either one of two selected quadrants (0..3, 4...7, 8...11 or 12...15). Control inputs A,B then determine the location within the quadrant.

Examples:

- From state ③, if C=High, go to ⑤ else go to ⑧
- From state ⑦, if C=High, go to ③ else stay in ⑦
- From state ⑨, unconditionally go to ②
- From state ⑥, execute the truth-table below

A B	C= Low	C=High
0 0	⑫	⓪
1 0	⑬	①
0 1	⑭	②
1 1	⑮	③



30 MHz State Machine, 16 States, 2-Way/8-Way Branch, 8 Outputs

CLB SWITCHING CHARACTERISTIC GUIDELINES

Description		Symbol	Speed Grade -50 Min	-50 Max	-70 Min	-70 Max	-100 Min	-100 Max	Units
Combinatorial Delay Logic Variables a, b, c, d, e, to outputs x, y	1	T_{ILO}		14		9		7	ns
Sequential delay Clock k to outputs x, y Clock k to outputs x,y when Q is returned through function generators F or G to drive x, y	8	T_{CKO}		12 23		8 15		7 12	ns ns
Set-up time before clock K Logic Variables a, b, c, d, e Data In di Enable Clock ec Reset Direct inactive rd	2 4 6	T_{ICK} T_{DICK} T_{ECCK}	12 8 10 2		8 5 7 1		7 4 5 1		ns ns ns ns
Hold Time after clock k Logic Variables a, b, c, d, e Data In di Enable Clock ec	3 5 7	T_{CKI} T_{CKDI} T_{CKEC}	0 6 0		0 4 0		0 2 0		ns ns ns
Clock Clock High time* Clock Low time* Max. flip-flop toggle rate*	11 12	T_{CH} T_{CL} F_{CLK}	9 9 50		7 7 70		5 5 100		ns ns MHz
Reset Direct (rd) rd width delay from rd to outputs x, y	13 9	T_{RPW} T_{RIO}	12	12	8	8	7	7	ns ns
Master Reset (MR) MR width delay from MR to outputs x, y		T_{MRW} T_{MRQ}	38	30	25	20	21	17	ns ns

BUFFER (Internal) SWITCHING CHARACTERISTIC GUIDELINES

Description	Symbol	Speed Grade -50 Min	-50 Max	-70 Min	-70 Max	-100 Min	-100 Max	Units
Global and Alternate Clock Distribution** Either: **Normal** IOB input pad to clock buffer input Or: **Fast** (CMOS only) input pad to clock buffer input Plus: Clock buffer input to any clock k	T_{PID} T_{PIDC}		9 5 9		6 3 6		4 2 5	ns ns ns
TBUF driving a Horizontal Longline (L.L.)** I to L.L. while T is Low (buffer active) T↓ to L.L. active and valid T↑ to L.L. (inactive) with single pull-up resistor with pair of pull-up resistors	T_{ID} T_{ON} T_{PUS} T_{PUF}		8 15 34 17		5 9 22 11		4 7 14 7	ns ns ns ns
BIDI Bi-directional buffer delay			6		4		3	ns

Autumn 1989 Users' Group Meetings

Starting in mid-October, Xilinx will be conducting a worldwide series of Users' Group meetings. These sessions are intended to provide attendees with in-depth training on the use of the new Xilinx Design Manager, version 2.21 of the DS23 Automated Design Implementation software, and version 3.0 of XACT. (The release of XACT version 3.0 is scheduled for October.) This training will help increase the productivity of designers using LCAs, so every Xilinx user is encouraged to attend. Although the seminars are targeted for current Xilinx users, new customers also are welcome. Each participant will receive the seminar notes and a copy of the new 1989 Programmable Gate Array DataBook.

Please contact your local Xilinx sales office or sales representative for information regarding the dates and locations of Xilinx Users' Group meetings in your area.

Autumn 1989 Users' Group Meeting Agenda

I. Introduction

II. Design Flow and the Xilinx Design Manager

III. Design Entry
Recommended Design Techniques

IV. Design Implementation
ADI ver. 2.21 Update
Merge-then-Map vs. Map-then-Merge Design Flow
User Control of Partitioning
New Features of APR
XACT 3.0 Update

V. Design Verification
XNF Conversion Programs

VI. Upcoming Product Preview

XILINX

2100 Logic Drive
San Jose, CA 95124-3450